

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
28 December 2000 (28.12.2000)

PCT

(10) International Publication Number
WO 00/79411 A2

(51) International Patent Classification⁷: G06F 17/00

(21) International Application Number: PCT/US00/15676

(22) International Filing Date: 7 June 2000 (07.06.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/337,172 21 June 1999 (21.06.1999) US

(71) Applicant: SUN MICROSYSTEMS, INC. [—/US]; 901 San Antonio Road, M/S: UPAL01-521, Palo Alto, CA 94303 (US).

(72) Inventors: UHLER, Stephen; Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 (US). DIGIORGIO, Rinaldo; 20 Mile Common Road, Easton, CT 06612 (US). BENDER, Michael; 155 Sunbeam Avenue, Boulder Creek, CA 95006 (US).

(74) Agents: HECKER, Gary, A. et al.; The Hecker Law Group, Suite 2300, 1925 Century Park East, Los Angeles, CA 90067 (US).

(81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

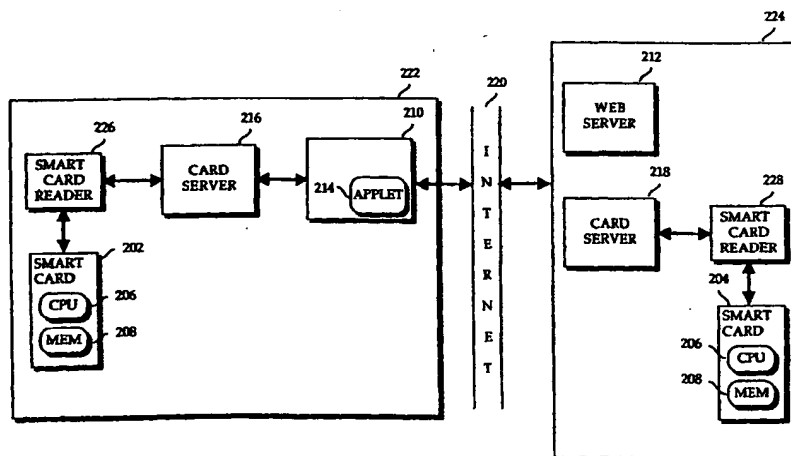
(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— Without international search report and to be republished upon receipt of that report.

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND APPARATUS FOR COMMERCIAL TRANSACTIONS VIA THE INTERNET



(57) Abstract: A method and apparatus for electronic commerce on the Internet is described. In one or more embodiments of the invention, electronic commerce transactions involving the transfer of funds stored on a smart card are accomplished via the Internet. According to an embodiment of the invention, an applet that is running within browser software interacts with a smart card server to access the software and data that resides on a smart card. The applet and smart card server communicate using the browser's mechanism for communicating via the Internet. Where, for example, the browser communicates over the Internet using the Hypertext Transport Protocol (HTTP), the applet and smart card server communicate using HTTP messages. The smart card server communicates with a smart card via the communications protocol of the smart card reader.

WO 00/79411 A2

METHOD AND APPARATUS FOR COMMERCIAL TRANSACTIONS VIA THE INTERNET

BACKGROUND OF THE INVENTION

5 1. FIELD OF THE INVENTION

This invention relates to electronic commerce and, more specifically, to commercial transactions via the Internet.

Portions of the disclosure of this patent document may contain material that is subject to copyright protection. The copyright owner has no objection to
10 the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the Patent and Trademark Office file or records, but otherwise reserves all copyright rights whatsoever. Sun, Sun Microsystems, the Sun logo, Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other
15 countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International in the United States and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

2. BACKGROUND ART

20 It is now possible to make purchases over the Internet (called "electronic commerce"), but there are risks of tampering, fraud, and theft in such transactions. These risks could be reduced or eliminated by using smart cards. However, current systems do not allow the best use of smart cards in electronic commerce.

The problem with the use of smart cards in current electronic commerce schemes can be understood by the following review of electronic commerce, electronic payment schemes, smart cards and the Internet.

Electronic Commerce

- 5 Electronic Commerce includes, for example, electronic banking and bill payment, as well as electronic purchases. Using a computer network, for example, commerce information is exchanged between a "server application" that provides the information or services, and a "client application" that receives the provided information or services.
- 10 Electronic commerce mechanisms that exist on the Internet allow users the ability to exchange commerce information. Commerce information exchange on the Internet is accomplished using software that transmits information between the server and the client to perform a transaction. The problem with current electronic commerce mechanisms is that electronic
- 15 commerce software is vulnerable to tampering, particularly where the software resides on a computer system that is accessible to the public. For example, kiosks are computer systems that typically unattended and allow a user to walk up and execute software that resides on the system. Where the kiosk is being used for electronic commerce, it is possible for someone to modify the electronic
- 20 commerce software that resides on the kiosk to capture another user's electronic commerce information (e.g., credit card information), for example.

Electronic Payment Schemes

- As with commercial transactions, most electronic commerce transactions involve a payment, or transfer, of funds. For example, when a client (the buyer)
- 25 wishes to purchase merchandise from an Internet web site (the seller or

merchant), the buyer selects the item from the seller's Web site and then selects from a set of payment options. If, for example, the buyer chooses to pay using a credit card, the buyer provides the credit card information. Once payment via the buyer's credit card is approved, the seller notifies the buyer and commences
5 delivery of the merchandise purchased by the buyer.

Electronic commerce transactions may involve the use of so called "digital wallet" software. A digital wallet has characteristics that imitate a physical wallet. Like a physical wallet, a digital wallet can hold different forms of payment (e.g., currency, credit cards and debit cards). While a physical wallet may have a user's
10 driver's license for identification, a digital wallet might include a digital certificate that can be used to identify (or authenticate) the user. In addition, a digital wallet can contain shipping information (e.g., the buyer's shipping address). The digital wallet may implement an encryption scheme as well for secure transmission of the digital wallet information.

15 One method that uses a digital wallet is digiCash (available from DigiCash BV). The buyer opens an account with an account provider such as a bank and downloads the digital wallet software. The buyer in conjunction with the bank creates "coins" (or electronic currency) that are digitally signed by the bank. The digital wallet can be used to exchange coins with other wallets. The coins can
20 also be presented to the bank to be cashed in for physical currency, for example.

A digital wallet that uses credit cards is available from CyberCash. A user downloads CyberCash's digital wallet and inputs credit card information into the digital wallet. The user also registers the credit card with CyberCash. The digital wallet registers itself with the user's Internet browser as a helper application.
25 When the user approves a transaction, an encrypted payment order is sent to

the merchant. The merchant adds some payment information, signs the order and forwards it to CyberCash.

A problem with the digital wallet scheme is that software and/or data for the digital wallet is stored on a computer that may be unsecured or unattended.

- 5 This makes it possible for unauthorized users to pretend they are legitimate users by using the computer to perform a transaction. Another problem is the possibility of obtaining software and data from an unsecured computer and using it to create fraudulent transactions.

Smart Cards

- 10 The problem of unsecured computers can be solved by the use of smart cards. Smart cards are credit card sized devices that can easily be secured by keeping them on your person.

- Smart cards can be used to transfer funds electronically. For example, a smart card might be inserted into a card reader of a pay phone to transfer funds
15 from the smart card to pay to, for example, the pay phone to pay for the card holder's telephone call.

- A smart card includes a microprocessor and memory. The smart card's microprocessor can be programmed to perform an electronic commerce transaction to transfer electronic funds that are stored in its memory. The
20 electronic funds that are stored in the smart card's memory can be obtained from the card issuer, for example.

- Electronic funds can also be transferred between smart cards. For example, electronic funds stored on a buyer's smart card can be transferred to a merchant's smart card to pay for the buyer's purchase. A single, integrated
25 software application is used to establish communication with both smart cards

simultaneously to perform the transaction (e.g., receive the electronic funds from the buyer, verify the electronic funds received from the buyer and transfer the funds to the merchant's card).

Problems exist with using a smart card over the Internet. Existing
5 software packages used to transfer smart card funds do not transmit messages formatted using the communication protocol adopted by most browsers (i.e., Hypertext Transfer Protocol, HTTP). This results in an inability to cross firewalls and other security mechanisms that limit passage based to HTTP messages. Further, the existing software packages do not allow an Internet user to use a
10 browser to transfer electronic funds stored on the smart card using the browser's communication protocols.

As discussed above, existing electronic commerce approaches used on the Internet (e.g., digital wallet software schemes) store the electronic funds and software (e.g., digital wallet software) in a computer system's storage. In
15 contrast to existing Internet electronic commerce approaches, both the electronic funds and the software to access the electronic funds can be stored on a smart card. The potential for tampering is reduced when a smart card is used, because it is easier to store the smart card in a secure place.

It would be beneficial to be able to use smart cards in electronic commerce
20 on the Internet. Further, it would be beneficial to be able to access a smart card from within a browser such that the electronic funds stored on the smart card can be transferred via the Internet.

A general description of the Internet followed by an overview of object-oriented programming and a review of the Java virtual machine's processing
25 environment are provided to better understand the problems associated with performing electronic commerce on the Internet.

Internet

The Internet is an example of a world wide communications network comprised of various physical networks that interconnect computing devices. The Internet is comprised of many physical networks that interconnect
5 computing devices. For example, a personal computer in a user's home can be connected via one or more networks that comprise the Internet to a computer system regardless of either's location to gain access to information that is resident on that computer system. A user's request can be transported via the Internet's networks to the computer system. A response from the computer
10 system can be transmitted to the user via the Internet.

The Transport Control Protocol/Internet Protocol (TCP/IP) are the basic communications protocol for transmitting information over Internet. A communications protocol typically defines the format for a packet, or bundle, of data that is to be transmitted. A packet usually includes control information
15 (e.g., destination, origin, packet length, etc.), the data to be transmitted and error detection and correction. Other communications protocols, such as Hypertext Transmission Protocol (HTTP) and File Transfer Protocol (FTP), are built on top of TCP/IP. Resources (e.g., servers, services, program code, and files) are accessible via the Internet and are typically referenced by a universal resource
20 locator (URL) that identifies the resource, the location of the resource and the protocol used to obtain the resource. A URL is mechanism by which a resource can be identified in a request. HTTP and FTP are mechanisms by which the request is communicated.

One example of a resource that can be requested by specifying a URL is a
25 Hypertext Markup Language (HTML) document that defines a page of graphic content. HTML is a language that can be used to specify the page's graphic user

interface (GUI) elements. An HTML document is transmitted via the HTTP communications protocol to a client that is running a software package referred to as a browser. A browser provides a GUI to display a page of information that is defined using HTML. The browser parses the HTML statements to generate
5 and display the page's GUI elements in the browser's display area. The browser further provides a mechanism for the user to input information and/or to submit a request which the browser forwards, via the Internet, to the appropriate Internet server using a communications protocol such as HTTP.

A communications network can be characterized as either an external
10 network (i.e., extranet) or an internal network (intranet). An extranet is a communications network that is considered to be external with reference to a given organization or entity. A network may be considered to be external simply because it is under another's administration and control. When viewed from a corporation's perspective, for example, the Internet is comprised of
15 networks that are examples of extranets.

Similarly, a communications network to which access is controlled or restricted is an internal network (or intranet). An intranet operates over a physical network that is under a given entity's administrative control.

An intranet can be connected to an extranet via a physical connection such
20 as a modem and telephone line. Routing hardware and/or software is used to route packets between the intranet and an extranet via a physical connection. A gateway which is comprised of hardware and/or software is typically used to act as an entrance and exit into a communications network. For example, an intranet can use a gateway through which packets directed to and from the
25 intranet must pass. A gateway can further perform conversions between otherwise incompatible communications networks.

An entity may wish to limit the packets that are allowed access to its intranet. For example, an entity may wish to limit entry to information that is resident on its intranet such that it is not accessible to extranet users (e.g., an Internet user unaffiliated with the entity).

- 5 A firewall is one example of a technique that implements a restrictive and controlled access approach to an intranet (e.g., between an intranet and an extranet). A firewall is hardware and/or software (typically considered part of a gateway) that examines packet data to determine whether the packet should be forwarded to/from the intranet. The firewall identifies the destination and/or
- 10 origin addresses to determine whether to forward the packet, for example. Where the firewall has been configured to stop the entry of packets from certain sources, the firewall examines the origin of the message and does not forward a message from an unauthorized source. If, for example, the firewall is configured to stop Internet packets from entering an intranet, the firewall blocks packets
- 15 whose origin is the Internet. Similarly, a firewall can be used to intercept and stop a packet that is destined for an unauthorized destination (e.g., an extranet).

- The restrictive and controlled access that is enforced by a firewall is advantageous because it reduces the chance of a security breach by an unauthorized user who otherwise might gain access to the intranet. However, a
- 20 firewall prohibits an intranet user from accessing the extranet via the intranet.

- To allow a user to gain access to the Internet from the intranet, a proxy server can be installed on the intranet which has access to both the intranet and the Internet. A proxy server acts as a proxy to forward requests on another's (e.g., an application's or user's) behalf. A proxy server forwards a message
- 25 without modifying its content.

A proxy server typically performs application-level filtering of messages. That is, a proxy server examines application-level messages to determine whether and to whom the message should be forwarded. A proxy server can be used, for example, to forward information between two applications (or users) 5 that reside on different intranets or between an intranet application (or user) and an extranet (e.g., the Internet). To access the Internet, for example, an intranet user sends a request directed to the Internet to the proxy server which forwards the request unchanged to the Internet.

Neither the firewall nor a proxy server allow access by an authorized user 10 attempting to gain access to the corporation's intranet from outside the intranet. The purpose of the firewall is to prohibit external access to the intranet. One purpose of a proxy server is to facilitate access to an extranet from within the intranet.

Object-Oriented Programming

15 Object-oriented programming is a method of creating computer programs by combining certain fundamental building blocks, and creating relationships among and between the building blocks. The building blocks in object-oriented programming systems are called "objects." A software application can be written using an object-oriented programming language 20 whereby the program's functionality is implemented using these objects.

An object is a programming unit that groups together a data structure (one or more instance variables) and the operations (methods) that can use or affect that data. Thus, an object consists of data and one or more operations or procedures that can be performed on that data. The joining of data and 25 operations into a unitary building block is called "encapsulation."

An object can be instructed to perform one of its methods when it receives a "message." A message is a command or instruction sent to the object to execute a certain method. A message consists of a method selection (e.g., method name) and a plurality of arguments. A message tells the receiving
5 object what operations to perform.

One advantage of object-oriented programming is the way in which methods are invoked. When a message is sent to an object, it is not necessary for the message to instruct the object how to perform a certain method. It is only necessary to request that the object execute the method. This greatly
10 simplifies program development.

Object-oriented programming languages are predominantly based on a "class" scheme. The class-based object-oriented programming scheme is generally described in Lieberman, "Using Prototypical Objects to Implement Shared Behavior in Object-Oriented Systems," OOPSLA 86 Proceedings,
15 September 1986, pp. 214-223.

An object class provides a definition for an object which typically includes both fields (e.g., variables) and methods. An object class is used to create a particular "object instance." (The term "object" by itself is often used interchangeably to refer to a particular class or a particular instance.) An
20 instance of an object class includes the variables and methods defined for that class. Multiple instances can be created from the same object class. Each instance that is created from the object class is said to be of the same type or class.

To illustrate, an employee object class can include "name" and "salary" instance variables and a "set_salary" method. Instances of the employee object
25 class can be created, or instantiated for each employee in an organization. Each object instance is said to be of type "employee." Each employee object instance

includes "name" and "salary" instance variables and the "set_salary" method. The values associated with the "name" and "salary" variables in each employee object instance contain the name and salary of an employee in the organization. A message can be sent to an employee's employee object instance to invoke the
5 "set_salary" method to modify the employee's salary (i.e., the value associated with the "salary" variable in the employee's employee object).

A hierarchy of classes can be defined such that an object class definition has one or more subclasses. A subclass inherits its parent's (and grandparent's etc.) definition. Each subclass in the hierarchy may add to or modify the
10 behavior specified by its parent class. The parent class is also referred to as a superclass with respect to its subclass(es). Some object-oriented programming languages support multiple inheritance where a subclass may inherit a class definition from more than one parent class. Other programming languages support only single inheritance, where a subclass is limited to inheriting the class
15 definition of only one parent class. The Java programming language also provides a mechanism known as an "interface" which comprises a set of constant and abstract method declarations. An object class can implement the abstract methods defined in an interface. Both single and multiple inheritance are available to an interface. That is, an interface can inherit an interface definition
20 from more than one parent interface.

Platform-Independent Programming Languages and Program Execution

Object-oriented software applications (e.g., applications written using the Java programming language) typically comprise one or more object classes and interfaces. Many programming languages can be used to write a program
25 which is compiled into machine-dependent (or platform-dependent), executable code. However, in other languages such as the Java programming language,

program code (e.g., classes) may be compiled into platform-independent
bytecode class files. Each class contains code and data in a platform-independent
format. A bytecode includes a code that identifies an instruction (an opcode) and
none or more operands to be used in executing the instruction. The computer
5 system acting as the execution vehicle contains a program such as a virtual
machine, which is responsible for executing the platform-independent code (e.g.,
bytecodes generated from a program written using the Java programming
language).

Platform-independent programs have an advantage of being usable on
10 multiple platforms. There is no need to develop program code for multiple
platforms. The same platform-independent program can be executed on
multiple platforms using a virtual machine or other mechanism that is
configured to translate the platform-independent code into platform-dependent
code. Thus, an application developer can develop one version of an application's
15 program code that can ultimately be executed on multiple platforms, for
example.

Applications may be designed as standalone applications, or as "applets"
which are identified by an applet tag in an HTML (Hypertext Markup Language)
document, and loaded by a browser application. The class files associated with
20 an application or applet may be stored on the local computing system, or on a
server accessible over a network. Each class file is loaded into the virtual
machine, as needed, by the "class loader."

The classes of an applet are loaded on demand from the network (stored
on a server), or from a local file system, when first referenced during the applet's
25 execution. The virtual machine locates and loads each class file, parses the class
file format, allocates memory for the class's various components, and links the

class with other already loaded classes. This process makes the code in the class readily executable by the virtual machine.

Figure 1 illustrates the development and runtime environments for a processing system. In the development environment, a software developer
5 creates class source files 100 which contain the programmer readable class definitions, including data structures, method implementations and references to other classes. Class source files 100 are provided to compiler 101, which compiles class source files 100 into compiled ".class" (or class) files 102 that contain bytecodes executable by a virtual machine. Class files 102 are stored
10 (e.g., in temporary or permanent storage) on a server, and are available for download over a network. Alternatively, class files 102 may be stored locally in a directory on the client platform.

The runtime environment contains a virtual machine (VM) 105 which is able to execute bytecode class files and execute native operating system ("O/S")
15 calls to operating system 109 when necessary during execution. Virtual machine 105 provides a level of abstraction between the machine independence of the bytecode classes and the machine-dependent instruction set of the underlying computer hardware 110, as well as the platform-dependent calls of operating system 109.

20 Class loader and bytecode verifier ("class loader") 103 is responsible for loading bytecode class files 102 and supporting class libraries 104 written using a programming language into virtual machine 105 as needed. Class loader 103 also verifies the bytecodes of each class file to maintain proper execution and enforcement of security rules. Within the context of runtime system 108, either
25 an interpreter 106 executes the bytecodes directly, or a "just-in-time" (JIT)

compiler 107 transforms the bytecodes into machine code, so that they can be executed by the processor (or processors) in hardware 110.

Native code, e.g., in the form of a linked library 111, is loaded when a class (e.g., from class libraries 104) containing the associated native method is
5 instantiated within the virtual machine. Linked library 111 can be, for example, a "shared object" library in the Solaris™ or UNIX environment that is written as a ".so" file, or linked library 111 may take the form of a dynamic loadable library written as a ".dll" file in a Windows environment.

Interpreter 106 reads, interprets and executes a bytecode instruction
10 before continuing on to the next instruction. JIT compiler 107 can translate multiple bytecode instructions into machine code that is then executed. Compiling the bytecodes prior to execution results in faster execution. If, for example, the same bytecode instruction is executed multiple times in a program's execution, it must be interpreted each time it is executed using
15 interpreter 106. If JIT compiler 107 is used to compile the program, the bytecode instruction may be translated once regardless of the number of times it is executed in the program. Further, if the compilation (i.e., output of JIT compiler 107) is retained, there is no need to translate each instruction during program execution.

SUMMARY OF THE INVENTION

A method and apparatus for electronic commerce on the Internet is described. In one or more embodiments of the invention, electronic commerce transactions involving the transfer of funds stored on a smart card are
5 accomplished via the Internet.

The invention provides a smart card server to handle communication between a smart card and the Internet. According to an embodiment of the invention, an applet that is running within browser software interacts with the smart card server to access the software and data that resides on a smart card.
10 The applet and smart card server communicate using the browser's mechanism for communicating via the Internet. Where, for example, the browser communicates over the Internet using the Hypertext Transfer Protocol (HTTP), the applet and smart card server communicate using HTTP messages. The smart card server communicates with a smart card via the communications protocol of
15 the smart card reader.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates the compile and runtime environments for a processing system.

Figure 2 provides an overview of an architecture for use in an electronic
5 commerce transaction using one or more smart cards according to an embodiment of the invention.

Figure 3 provides an example of communications between smart cards using an applet and card servers according to an embodiment of the invention.

Figure 4 provides an overview of smart card authentication according to
10 one or more embodiments of the invention.

Figure 5 is a block diagram of one embodiment of a computer system capable of providing a suitable execution environment for an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

A method and apparatus for commercial transactions via the Internet is described. In the following description, numerous specific details are set forth in order to provide a more thorough description of the present invention. It will
5 be apparent, however, to one skilled in the art, that the present invention may be practiced without these specific details. In other instances, well-known features have not been described in detail so as not to obscure the invention.

The invention uses a smart card server to handle communication between the smart card and the Internet. According to an embodiment of the
10 invention, an applet that is running within a browser interacts with a smart card server to access the software and data that resides on a smart card. The applet and smart card server communicate using the same communications protocol that a browser uses to communicate via the Internet. Where, for example, the browser communicates over the Internet using the Hypertext Transfer Protocol
15 (HTTP), the applet and smart card server communicate using HTTP messages. The smart card server communicates with a smart card via the communications protocol of the smart card reader.

Figure 2 provides an overview of an architecture for use in an electronic commerce transaction using one or more smart cards according to an
20 embodiment of the invention. In the example of Figure 2, electronic funds are to be transferred from smart card 202 to smart card 204. Smart cards 202 and 204 can be, Java Cards from Sun Microsystems, Inc. or smart cards from Mondex International, for example.

The holder of smart card 202 (i.e., the buyer) has identified a item for
25 purchase from a merchant's site (i.e., computer system 224) using browser 210 executing on computer system 222. That is, the buyer navigates Internet 220

using browser 210 to access a merchant's Web server 212. Web server 212 responds to a uniform resource locator (URL) request for a Web page sent by browser 210 for the buyer, for example. The Web page that is sent may include items that can be purchased by the buyer. The buyer selects the desired item
5 and clicks on a button to indicate a desire to purchase the selected item.

Applet 214 executes in browser 210. In an embodiment of the invention, applet 214 may comprise one or more object classes written using the Java programming language. However, it should be apparent that applet 214 can be written using other programming languages. Applet 214 can be downloaded
10 from the merchant's Web server 212 or elsewhere, for example.

Applet 214 acts as an interface between card servers 216 and 218. Card servers 216 and 218 act as interfaces between applet 214 and smart cards 202 and 204 via smart card readers 226 and 228 (respectively). Information is transmitted between smart cards 202 and 204 via card servers 216 and 218 and applet 214.

15 Card servers 216 and 218 communicate with smart cards 202 and 204 using the communications protocol of smart card readers 226 and 228 (e.g., a serial communications protocol) and the protocol of smart cards 216 and 218 (e.g., a smart card command protocol). Further, card servers 216 and 218 communicate with applet 214 using the communications protocol that is used by
20 browser 210 (e.g., the HTTP protocol).

Card servers 216 and 218 present an interface to browser 210 and applet 214 that resembles a set of Web pages. That is, applet 214 sends HTTP requests to card servers 216 and 218 that contain information intended for smart cards 202 and 204. Card servers 216 and 218 send Web page definitions that contain a
25 response from smart cards 202 and 204 to applet 214.

Further, card servers 216 and 218 provide a mechanism for applet 214 to communicate with smart cards 202 and 204. For security reasons, browser 210 does not allow an applet (e.g., applet 214) that is running within browser 210 to directly communicate with devices that reside on the platform such as smart card reader 226 and smart card 202. Card server 216 provides a mechanism whereby
5 applet 214 can execute within browser 210 and be able to communicate with smart card 202 via smart card reader 226.

Card servers 216 and 218 encapsulate data intended for smart cards 202 and 204 into HTTP messages that can be forwarded via Internet 220 to the other
10 smart card(s). The HTTP messages that are generated by card servers 216 and 218 can be forwarded via Internet 220 through one or more Web proxies and firewalls that may otherwise restrict passage of non-HTTP messages. Thus, embodiments of the invention can be used to facilitate smart card transactions via Internet 220 across one or more web servers, Web proxies and firewalls.

Figure 3 provides an example of communications between smart cards
15 202 and 204 using applet 214 and card servers 216 and 218 according to an embodiment of the invention. In this example, applet 214 provides an interface for the buyer to enter payment information such as a personal identification number (or PIN) and the amount of the purchase. Applet 214 generates HTTP
20 message 302 (e.g., a URL request) that is directed to card server 216. HTTP message 302 can be, for example, of the form:

`http://buyerCardServer.com/verifyPin?form data`

The form data in HTTP message 302 comprises the PIN and purchase amount information entered by the buyer, for example. The verifyPin portion
25 of HTTP message 302 identifies the operation that is to be performed by smart

card 202. Card server 216 is identified in HTTP message 302 using card server 216's URL (e.g., buyerCardServer.com).

Card server 216 extracts the form data from HTTP message 302 and forwards the information that is needed by smart card 202 in card request 304 to smart card 202 via smart card reader 226 (not shown). Smart card 202 generates a response, card response 306, that is forwarded to card server 216 via smart card reader 226. Card server 216 packages the response received from smart card 202 in HTTP message 308 and forwards HTTP message 308 to applet 214 as a response to HTTP message 302.

HTTP message 308 can be an HTML Web page definition that contains the response received from smart card 202, for example. HTTP message 308 further contains information that identifies the state of the transaction. The state of the transaction can be used by card servers 216 and 218 to determine what request to make of smart cards 202 and 204. Further, applet 214 examines the state of the transaction. If the transaction state does not indicate a finished state for the transaction (e.g., a successful completion or an error), applet 214 forwards HTTP message 308 as HTTP message 310 to card server 218. As with HTTP message 302, HTTP message 310 identifies the URL of card server 218, an operation and form data. Card server 218 forwards the data contained in HTTP message 310 along with a command for processing the data as card request 312 to smart card 204 via smart card reader 228 (not shown).

Smart card 204 processes the data and transmits the result to card server 218 as card response 314. Card server 218 generates HTTP message 316 which is transmitted to applet 214 in response to HTTP message 310.

Applet 214 examines HTTP message 316 to determine the state of the electronic commerce transaction. If the transaction is not finished, the process

continues with applet 214 forwarding HTTP message 316 to card server 216 in a manner similar to that described with reference to HTTP message 302 above.

Smart Card Protocols

Different types of smart cards may use different protocols that may
5 require a different number of steps to perform an electronic commerce transaction. That is, one smart card protocol may require five steps while another may require twenty steps. The steps of an electronic commerce transaction correspond to states of the transaction in one embodiment of the invention.

10 To accommodate each smart card type, card servers 216 and 218 may include interface (or communication) software modules (e.g., program code) to accommodate different types of smart cards and their protocols. Each interface module communicates with a particular type of smart card (e.g., Java Card and/or Mondex smart card). For example, each interface module is capable of
15 generating messages to prompt the smart card to process the data included in the message in the desired manner for the current state of the transaction. It is not necessary for applet 214 to be aware of the specific interfaces needed to communicate with a particular smart card. Thus, it is not necessary to modify applet 214 when a new type of smart card is introduced. It is only necessary to
20 incorporate a new interface module for a new smart card into the card server (e.g., card servers 216 and 218).

In one embodiment of the invention, cards requests 304 and 312 and card responses 306 and 314 are communicated using a set of message types, or command protocol. For example, commands such as putString, getString and
25 getNext can be used in an embodiment of the invention. In this embodiment, card requests 304 and 312 include a putString message that transmits the data to

be processed to smart cards 202 and 204 (respectively). A getNext message is sent by card servers 216 and 218 to request that smart cards 202 and 204 (respectively) process the data that was sent in the putString message. A getString message fetches the information from smart cards 202 and 204 that
5 card servers 216 and 218 determine is needed to be sent to the other card given the current state of the transaction. Thus, a getString message may be used to retrieve the result calculated by a smart card from the data contained in a putString message.

As described above, HTTP messages 302, 308, 310 and 314 include
10 commands that identify the type of operations to be performed by smart cards 202 and 204. One example provided above is verifyPin which might be used to verify a PIN that is entered by the buyer. Other examples of commands that may be used in embodiments of the invention include: checkPin, setPin, changePin, resetPin, resetLog, generateResetLogCode, generateResetPinCode
15 and valueTransfer. The checkPin, setPin, changePin, resetPin and generateResetPinCode involve operations that modify PIN information. The resetLog and generateResetLogCode commands can be used to maintain an electronic commerce transaction log. The valueTransfer command may be used to transfer electronic funds from one smart card to another.

20 Authentication

One or more embodiments of the invention can be used to authenticate a user using smart cards. Authentication can comprise one or more of the steps in the course of an electronic commerce transaction, for example. Authentication may also be used for other purposes such as in determining whether a smart
25 card holder is permitted to access capabilities behind a firewall.

The smart cards that are used in authentication share a secret value that can be used to generate an authentication value and to verify the authenticity of an authentication value. The same secret is used in both the generation and the verification processes. Smart cards (e.g., Java Cards and/or Mondex smart cards) include authentication and/or verification program code that performs a set of operations which are proprietary to the card that are used to generate or verify an authentication value.

Generally, the secret value is used in conjunction with the authentication program code in a first smart card to generate an authentication value. The authentication value is verified using a second smart card's verification program code in conjunction with the same secret.

Figure 4 provides an overview of smart card authentication according to one or more embodiments of the invention. In the example of Figure 4, the authentication process is used to restrict access to computer resources. Client 400 includes browser 210, card server 216 and smart card 202. Server 402 includes computing resources that are access-restricted. Server 402 may act as a gateway to an intranet, for example. Reverse proxy 404 may be used in one or more embodiments of the invention to enforce a plurality of access policies. Reverse proxy 404 may be used to restrict access to the intranet, for example. Reverse proxy 404 can refuse access by an unauthenticated user and/or refuse access by an authenticated user to one or more resources where the authenticated user has insufficient access privileges for these resources, for example.

To access the resources of server side 402, an unauthenticated user using browser 210 contacts reverse proxy 404 (e.g., login request 420) to login to the intranet. Reverse proxy 404 generates response 422 which includes a login Web

page definition and applet 214. The login Web page prompts the user to insert a smart card into the smart card reader (not shown) attached to client 400 and enter the information needed to authenticate and/or identify the user. For example, the user may be prompted to enter the user's identification (userid) and password (e.g., PIN), for example.

Applet 214 generates HTTP message 424 that contains the userid and PIN and forwards HTTP message 424 to card server 216. Card server 216 generates card message 426 that contains the information (e.g., the user's PIN) needed to generate the authentication value. Card message 426 is forwarded to smart card 102 by card server 216. Smart card 202 performs calculations on the data to generate the authentication value using the secret value. Card message 428 includes the authentication value which is transmitted to card server 216.

Card server 216 generates HTTP message 430 that includes the authentication value and the user's userid. Card server 216 forwards HTTP message 430 to applet 214 which forwards it as HTTP message 434 to reverse proxy 404. Reverse proxy 404 forwards HTTP message 434 to card server 218. Card server 218 retrieves the authentication value from HTTP message 434 and generates card message 436 which is transmitted to smart card 204. Smart card 204 performs a verification operation on the authentication value and returns the result (success or failure) as card message 438.

Where the verification is successful, card server 218 retrieves access privileges 442 associated with the user's userid (e.g., userid 440) from access database 406. Card server 218 forwards the user's access privileges to reverse proxy 404 via HTTP message 444. Where the verification is unsuccessful, card server 218 forwards the unsuccessful status to reverse proxy 404 via HTTP message 444.

Embodiment of Computer Execution Environment (Hardware)

An embodiment of the invention can be implemented as computer software in the form of computer readable code executed on a general purpose computer such as computer 200 illustrated in Figure 2, or in the form of
5 bytecode class files executable within a Java runtime environment running on such a computer, or in the form of bytecodes running on a processor (or devices enabled to process bytecodes) existing in a distributed environment (e.g., one or more processors on a network). A keyboard 210 and mouse 211 are coupled to a system bus 218. The keyboard and mouse are for introducing user input to the
10 computer system and communicating that user input to processor 213. Other suitable input devices may be used in addition to, or in place of, the mouse 211 and keyboard 210. I/O (input/output) unit 219 coupled to system bus 218 represents such I/O elements as a printer, A/V (audio/video) I/O, etc.

Computer 200 includes a video memory 214, main memory 215 and mass
15 storage 212, all coupled to system bus 218 along with keyboard 210, mouse 211 and processor 213. The mass storage 212 may include both fixed and removable media, such as magnetic, optical or magnetic optical storage systems or any other available mass storage technology. Bus 218 may contain, for example, thirty-two address lines for addressing video memory 214 or main memory 215.
20 The system bus 218 also includes, for example, a 64-bit data bus for transferring data between and among the components, such as processor 213, main memory 215, video memory 214 and mass storage 212. Alternatively, multiplex data/address lines may be used instead of separate data and address lines.

In one embodiment of the invention, the processor 213 is a
25 microprocessor manufactured by Sun Microsystems, Inc., such as the SPARC™ microprocessor, or a microprocessor manufactured by Motorola, such as the

680X0 processor, or a microprocessor manufactured by Intel, such as the 80X86, or Pentium processor. However, any other suitable microprocessor or microcomputer may be utilized. Main memory 215 is comprised of dynamic random access memory (DRAM). Video memory 214 is a dual-ported video random access memory. One port of the video memory 214 is coupled to video amplifier 216. The video amplifier 216 is used to drive the cathode ray tube (CRT) raster monitor 217. Video amplifier 216 is well known in the art and may be implemented by any suitable apparatus. This circuitry converts pixel data stored in video memory 214 to a raster signal suitable for use by monitor 217. Monitor 217 is a type of monitor suitable for displaying graphic images.

Computer 200 may also include a communication interface 220 coupled to bus 218. Communication interface 220 provides a two-way data communication coupling via a network link 221 to a local network 222. For example, if communication interface 220 is an integrated services digital network (ISDN) card or a modem, communication interface 220 provides a data communication connection to the corresponding type of telephone line, which comprises part of network link 221. If communication interface 220 is a local area network (LAN) card, communication interface 220 provides a data communication connection via network link 221 to a compatible LAN. Wireless links are also possible. In any such implementation, communication interface 220 sends and receives electrical, electromagnetic or optical signals which carry digital data streams representing various types of information.

Network link 221 typically provides data communication through one or more networks to other data devices. For example, network link 221 may provide a connection through local network 222 to local server computer 223 or to data equipment operated by an Internet Service Provider (ISP) 224. ISP 224 in turn provides data communication services through the world wide packet data

communication network now commonly referred to as the "Internet" 225. Local network 222 and Internet 225 both use electrical, electromagnetic or optical signals which carry digital data streams. The signals through the various networks and the signals on network link 221 and through communication
5 interface 220, which carry the digital data to and from computer 200, are exemplary forms of carrier waves transporting the information.

Computer 200 can send messages and receive data, including program code, through the network(s), network link 221, and communication interface 220. In the Internet example, remote server computer 226 might transmit a
10 requested code for an application program through Internet 225, ISP 224, local network 222 and communication interface 220.

The received code may be executed by processor 213 as it is received, and/or stored in mass storage 212, or other non-volatile storage for later execution. In this manner, computer 200 may obtain application code in the
15 form of a carrier wave.

Application code may be embodied in any form of computer program product. A computer program product comprises a medium configured to store or transport computer readable code, or in which computer readable code may be embedded. Some examples of computer program products are CD-ROM
20 disks, ROM cards, floppy disks, magnetic tapes, computer hard drives, servers on a network, and carrier waves.

The computer systems described above are for purposes of example only. An embodiment of the invention may be implemented in any type of computer system or programming or processing environment.

Thus, a method and apparatus for commercial transactions via the Internet has been described in conjunction with one or more specific embodiments. The invention is defined by the claims and their full scope of equivalents.

CLAIMS

1. In a computer network, a method of performing an electronic commerce transaction comprising:
 - transmitting a first plurality of smart card messages between a first smart
 - 5 card and a first card server;
 - transmitting a first plurality of network messages between said first card server and program code executing within a browser;
 - transmitting said plurality of network messages to a second card server;
 - transmitting a second plurality of smart card messages between said
 - 10 second card server and a second smart card, said second plurality of smart card messages using said smart card protocol.
2. The method of claim 1 wherein said transmitting a first plurality of smart card messages between a first smart card and a first card server further comprises:
 - 15 said first smart card transmitting said first plurality of smart card messages to a first card reader:
 - said first card reader forwarding said first plurality of smart card messages to said first card server.
3. The method of claim 1 wherein said first plurality of smart card
- 20 messages use a smart card message protocol.

4. The method of claim 1 wherein said transmitting a first plurality of network messages between said first card server and program code executing within a browser further comprises:

- translating said first plurality of smart card messages into said first
5 plurality of network messages;
said first card server transmitting said first plurality of network messages to said program code.

5. The method of claim 1 wherein said plurality of network messages use a protocol of said computer network.

- 10 6. The method of claim 5 wherein said protocol of said computer network is Hypertext Transmission Protocol (HTTP).

7. The method of claim 1 wherein said program code is an applet written in the Java programming language.

8. A system comprising:
15 program code executing in a computer, said program code configured to send and receive network messages;
a first card server coupled to said program code, said first card server configured to send messages to and receive messages from a first smart card;
a second card server coupled to said program code, said second card
20 server configured to send messages to and receive messages from a second smart card.

9. The system of claim 8 wherein said first card server is coupled to said first smart card via a first card reader, said first card reader configured to forward messages between said first card server and said first smart card.

10. The system of claim 8 wherein said second card server is coupled to said second smart card via a second card reader, said second card reader configured to forward messages between said second card server and said second smart card.
- 5 11. The system of claim 8 wherein said program code is a browser running an applet.
12. The system of claim 11 wherein said applet is written using the Java programming language.
- 10 13. The system of claim 11 wherein said second card server coupled to said program code via a proxy server configured to enforce a plurality of access policies.

14 A computer program product comprising:

a computer usable medium having computer readable program code embodied therein configured to perform an electronic commerce transaction comprising:

5 computer readable program code configured to cause a computer to transmit a first plurality of smart card messages between a first smart card and a first card server;

computer readable program code configured to cause a computer to transmit a first plurality of network messages between said first card server and

10 program code executing within a browser;

computer readable program code configured to cause a computer to transmit said plurality of network messages to a second card server;

computer readable program code configured to cause a computer to transmit a second plurality of smart card messages between said second card
15 server and a second smart card, said second plurality of smart card message using said smart card protocol.

15. The computer program product of claim 14 wherein said computer readable program code configured to cause a computer to transmit a first plurality of smart card messages between a first smart card and a first card
20 server further comprises:

computer readable program code configured to cause said first smart card to transmit said first plurality of smart card messages to a first card reader:

computer readable program code configured to cause said first card reader to forward said first plurality of smart card messages to said first card
25 server.

16. The computer program product of claim 14 wherein said first plurality of smart card messages use a smart card message protocol.

17. The computer program product of claim 14 wherein said computer readable program code configured to cause a computer to transmit a first
5 plurality of network messages between said first card server and program code executing within a browser further comprises:

computer readable program code configured to cause a computer to translate said first plurality of smart card messages into said first plurality of network messages;

10 computer readable program code configured to cause said first card server to transmit said first plurality of network messages to said program code.

18. The computer program product of claim 14 wherein said plurality of network messages use a protocol of said computer network.

19. The computer program product of claim 18 wherein said protocol
15 of said computer network is Hypertext Transmission Protocol (HTTP).

20. The computer program product of claim 14 wherein said program code is an applet written in the Java programming language.

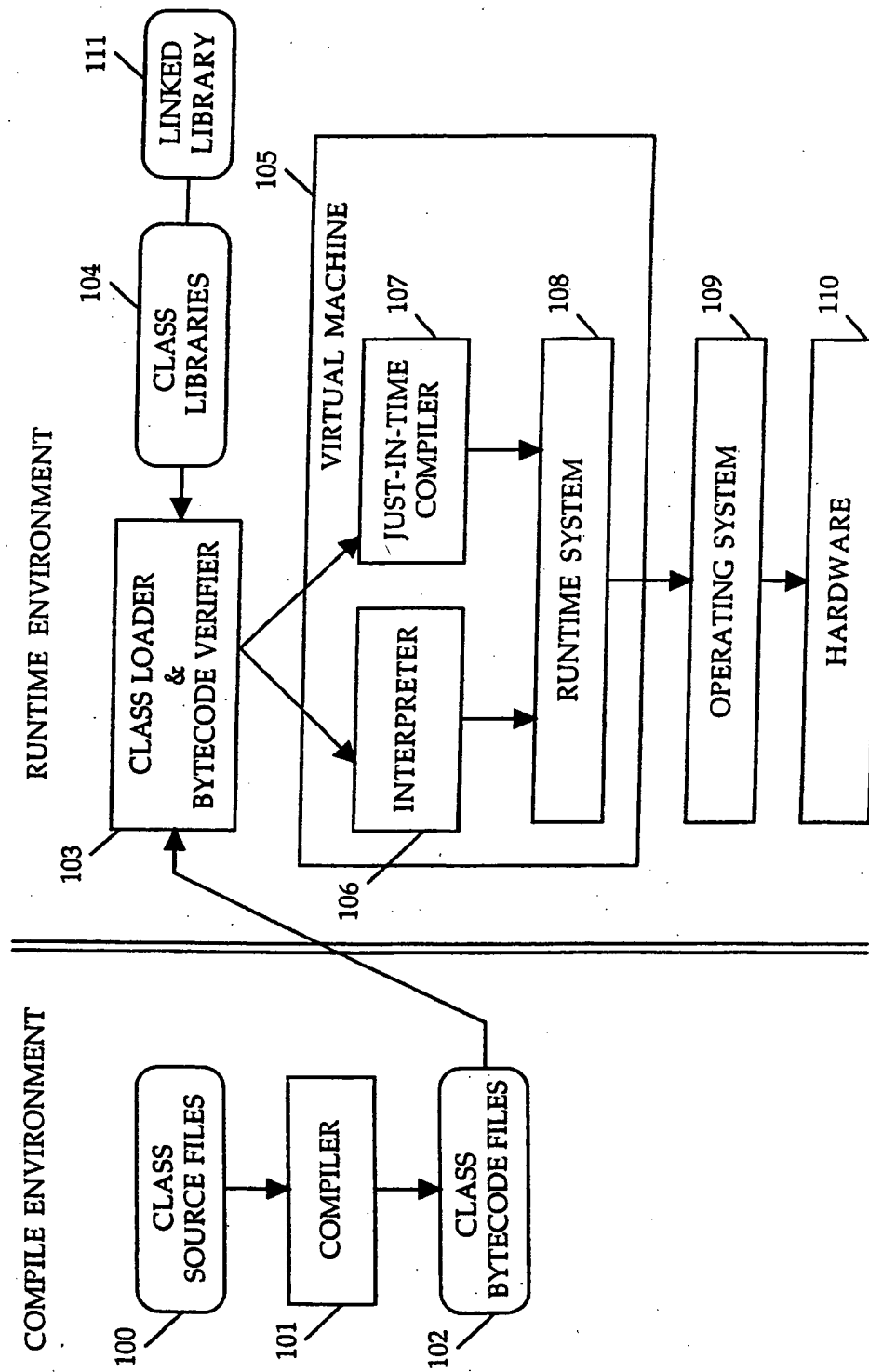


FIGURE 1

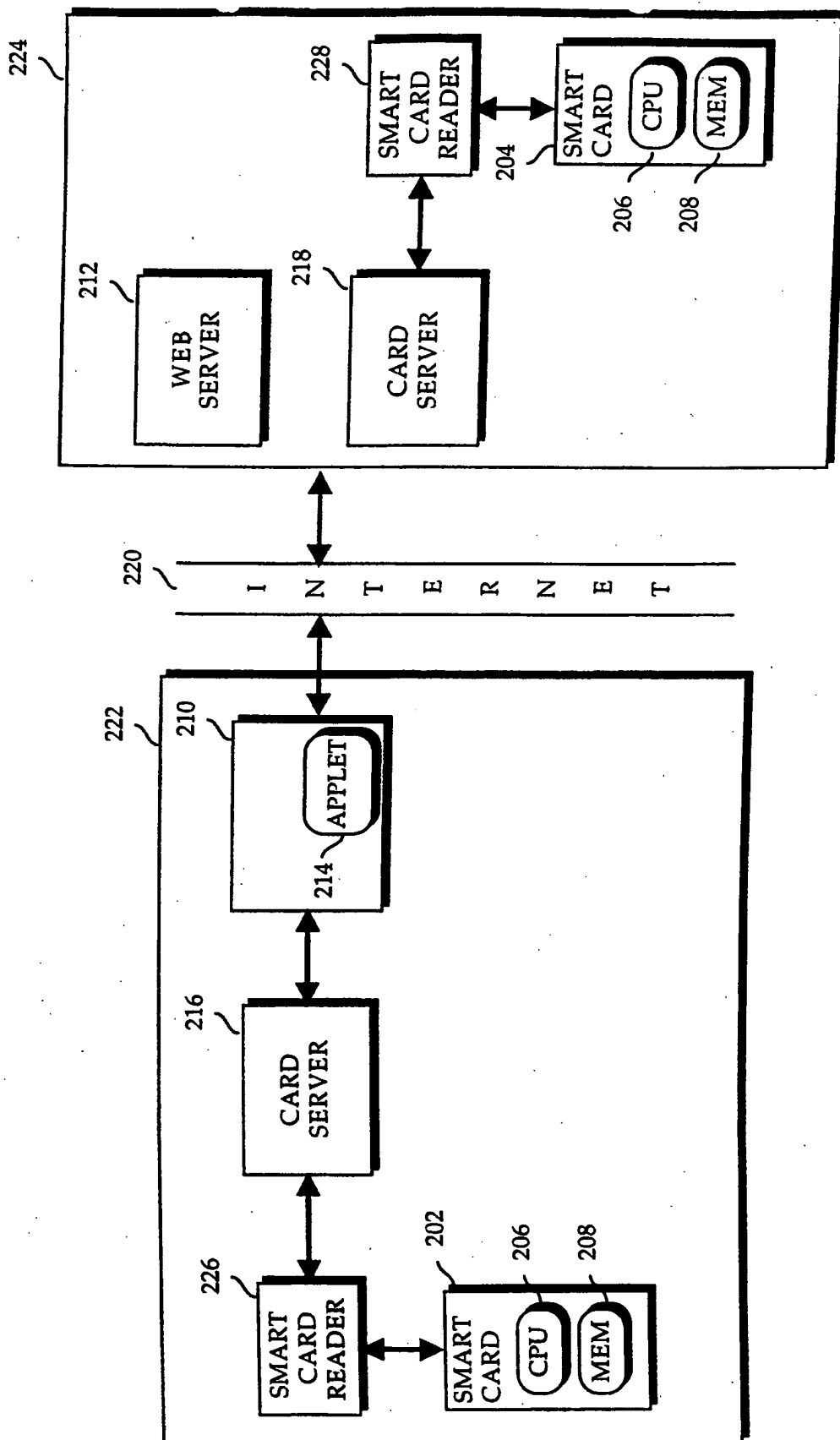


Figure 2

3/5

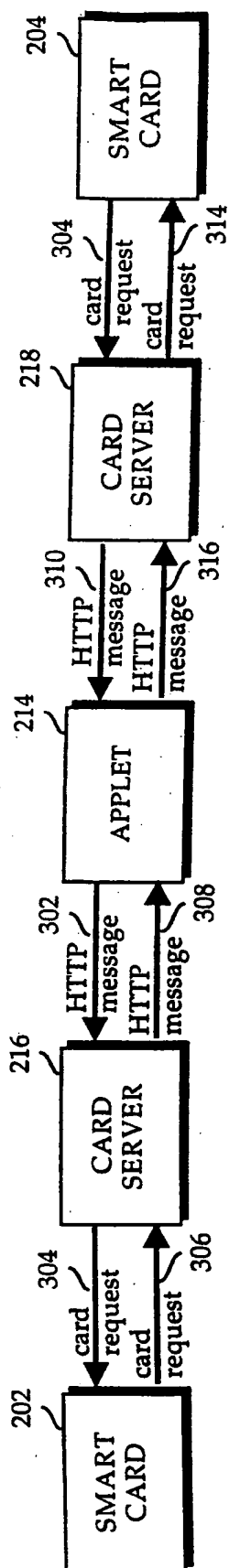


Figure 3

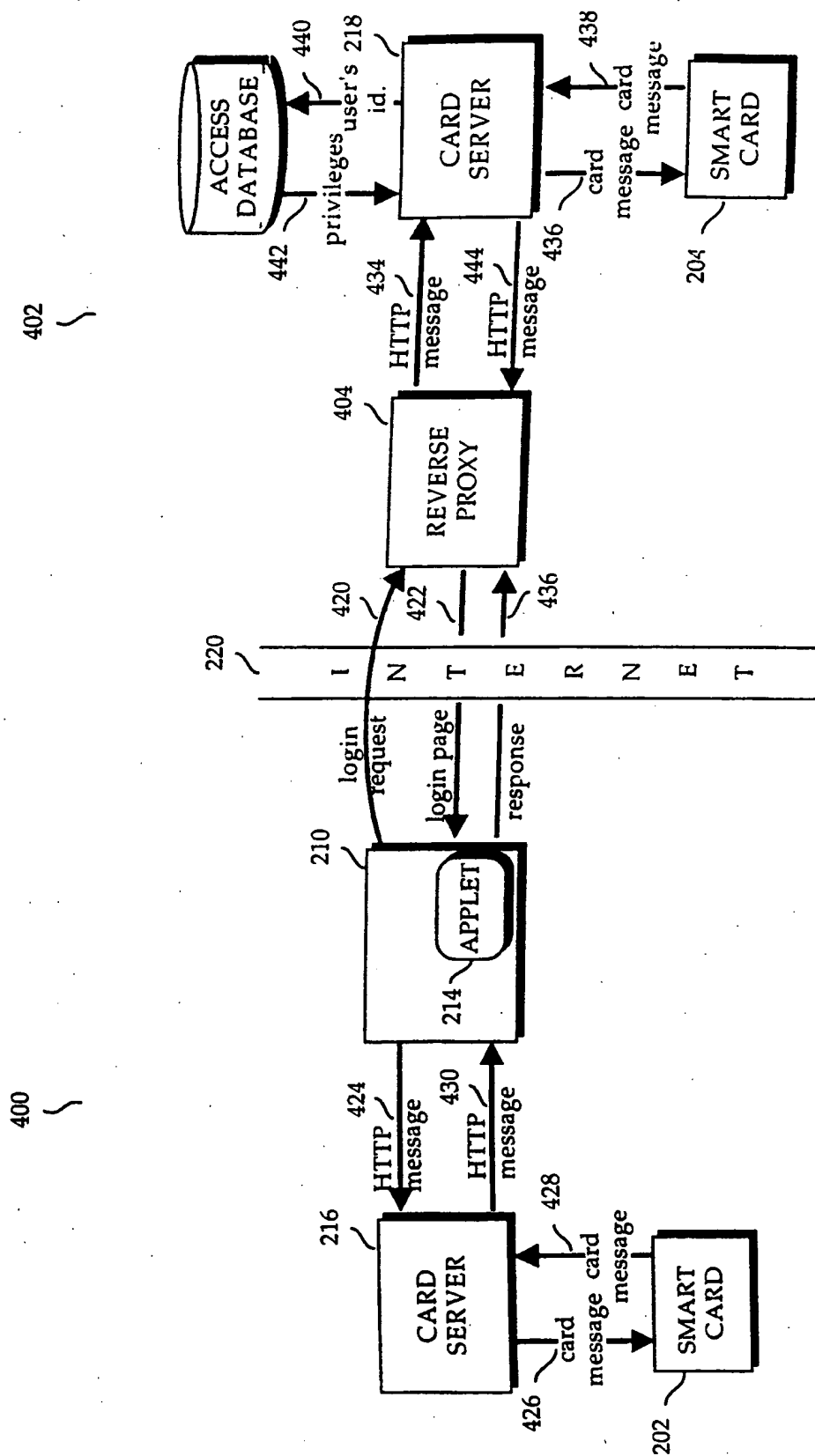
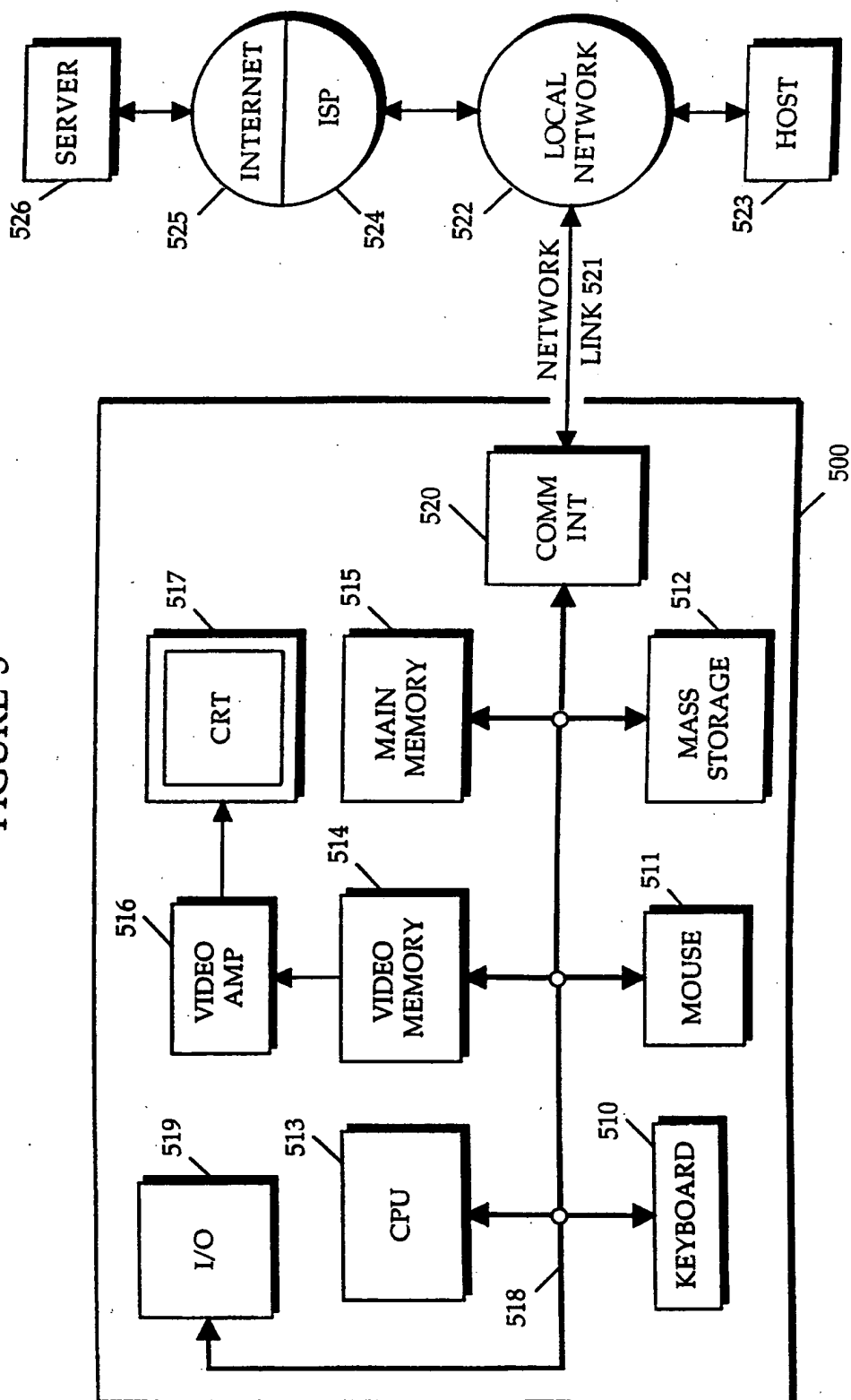


Figure 4

FIGURE 5



CORRECTED VERSION

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
28 December 2000 (28.12.2000)

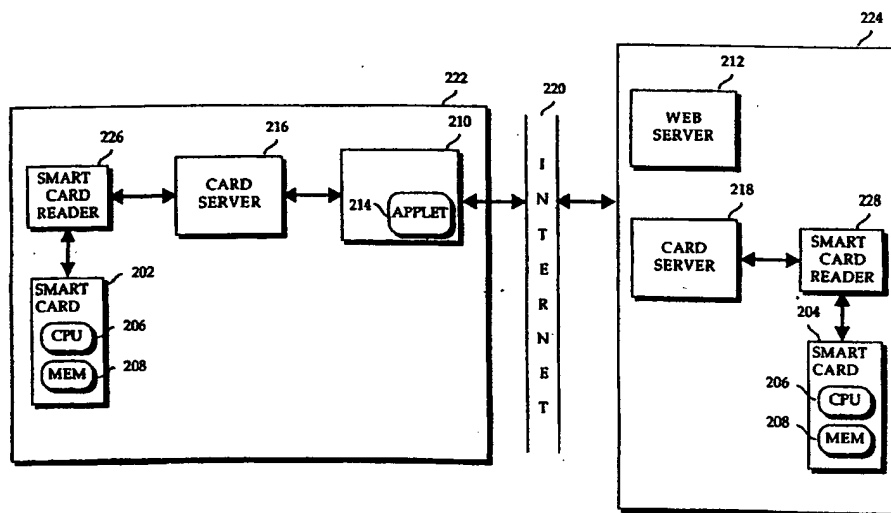
PCT

(10) International Publication Number
WO 00/79411 A2

- (51) International Patent Classification⁷: G06F 17/00 (74) Agents: HECKER, Gary, A. et al.; The Hecker Law Group, Suite 2300, 1925 Century Park East, Los Angeles, CA 90067 (US).
- (21) International Application Number: PCT/US00/15676
- (22) International Filing Date: 7 June 2000 (07.06.2000) (81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data: 09/337,172 21 June 1999 (21.06.1999) US
- (71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, M/S: UPAL01-521, Palo Alto, CA 94303 (US).
- (72) Inventors: UHLER, Stephen; Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 (US). DI-GIORGIO, Rinaldo; 20 Mile Common Road, Easton, CT 06612 (US). BENDER, Michael; 155 Sunbeam Avenue, Boulder Creek, CA 95006 (US).
- (84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).
- Published:
— Without international search report and to be republished upon receipt of that report.

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR COMMERCIAL TRANSACTIONS VIA THE INTERNET



(57) Abstract: A method and apparatus for electronic commerce on the Internet is described. In one or more embodiments of the invention, electronic commerce transactions involving the transfer of funds stored on a smart card are accomplished via the Internet. According to an embodiment of the invention, an applet that is running within browser software interacts with a smart card server to access the software and data that resides on a smart card. The applet and smart card server communicate using the browser's mechanism for communicating via the Internet. Where, for example, the browser communicates over the Internet using the Hypertext Transport Protocol (HTTP), the applet and smart card server communicate using HTTP messages. The smart card server communicates with a smart card via the communications protocol of the smart card reader.



(48) Date of publication of this corrected version:

21 June 2001

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(15) Information about Correction:

see PCT Gazette No. 25/2001 of 21 June 2001, Section II